Electronic Colloquium on Computational Complexity, Report No. 73 (2017)

New Insights on the (Non)-Hardness of Circuit Minimization and Related Problems*

Eric Allender[1] and Shuichi Hirahara[2]

1 **Department of Computer Science, Rutgers University, Piscataway, NJ, USA,** allender@cs.rutgers.edu
2 **Department of Computer Science, The University of Tokyo, Tokyo, Japan,** hirahara@is.s.u-tokyo.ac.jp

Abstract

The Minimum Circuit Size Problem (MCSP) and a related problem (MKTP) that deals with time-bounded Kolmogorov complexity are prominent candidates for NP-intermediate status. We show that, under very modest cryptographic assumptions (such as the existence of one-way functions), the problem of approximating the minimum circuit size (or time-bounded Kolmogorov complexity) within a factor of $n^{1-o(1)}$ is *indeed* NP-intermediate. To the best of our knowledge, these problems are the first natural NP-intermediate problems under the existence of an arbitrary one-way function.

We also prove that MKTP is hard for the complexity class DET under non-uniform NC^0 reductions. This is surprising, since prior work on MCSP and MKTP had highlighted weaknesses of "local" reductions such as $\leq_{\mathrm{m}}^{\mathsf{NC}^0}$. We exploit this local reduction to obtain several new consequences:

- MKTP is not in $\mathsf{AC}^0[p]$.
- Circuit size lower bounds are equivalent to hardness of a relativized version MKTP^A of MKTP under a class of uniform AC^0 reductions, for a large class of sets A.
- Hardness of MCSP^A implies hardness of MKTP^A for a wide class of sets A. This is the first result directly relating the complexity of MCSP^A and MKTP^A, for any A.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases computational complexity, Kolmogorov complexity, Circuit size

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

The Minimum Circuit Size Problem (MCSP) has attracted intense study over the years, because of its close connection with the natural proofs framework of Razborov and Rudich [32], and because it is a prominent candidate for NP-intermediate status. It has been known since [25] that NP-intermediate problems exist, if $\mathsf{P} \neq \mathsf{NP}$, but "natural" candidates for this status are rare. Problems such as factoring and Graph Isomorphism are sometimes put forward as candidates, but there are not strong complexity-theoretic arguments for why these problems should not lie in P. We prove that a very weak cryptographic assumption implies that a $n^{1-o(1)}$ approximation for MCSP is NP-intermediate.

MCSP is hard for SZK [8] under BPP reductions, but the situation is quite different, when more restricted notions of reducibility are considered. Recent results [20, 27, 9] have

* Supported by NSF grant CCF-1555409 (Allender)

ISSN 1433-8092

suggested that MCSP might not even be hard for P under logspace reductions (although the evidence is still inconclusive).

The input to MCSP consists of a pair (T, s), where T is a bit string of length 2^m representing the truth-table of an m-variate Boolean function, and $s \in \mathbb{N}$; $(T, s) \in \mathsf{MCSP}$ if there is a circuit computing T having size at most s. Note that, for different models of circuit (type of gates, allowable fan-in, etc.) and different measures of size (number of gates, number of wires, size of the description of the circuit, etc.) the resulting MCSP problems might have different complexity. No efficient reduction is known between different variants of the problem. However, all prior work on MCSP (such as [23, 4, 6, 27, 8, 34, 9, 20]) applies equally well to any of these variants. MCSP is also closely related to a type of time-bounded Kolmogorov complexity known as KT, which was defined in [4]. The problem of determining KT complexity, formalized as the language $\mathsf{MKTP} = \{(x, s) : \mathrm{KT}(x) \leq s\}$ has often been viewed as just another equivalent "encoding" of MCSP in this prior work. (In particular, our results mentioned in the paragraphs above apply also to MKTP.) Recently, however, some reductions were presented that are not currently known to apply to MCSP [5].

In this section, we outline the ways in which this paper advances our understanding of MCSP and related problems, while reviewing some of the relevant prior work.

Hardness is equivalent to circuit size lower bounds. Significant effort (e.g. [23, 27, 9, 20])) has been made in order to explain why it is so difficult to show NP-hardness of MCSP or MKTP. Most of the results along this line showed implications from hardness of MCSP to circuit size lower bounds: If MCSP or MKTP is NP-hard under some restricted types of reductions, then a circuit size lower bound (which is quite difficult to obtain via current techniques of complexity theory) follows. For example, if MCSP or MKTP is hard for TC^0 under Dlogtime-uniform $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$ reductions, then $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ and $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}(2^{\epsilon n})$ [27, 9].

Murray and Williams [27] asked if, in general, circuit lower bounds imply hardness of the circuit minimization problems. We answer their questions affirmatively in certain settings: A stronger lower bound $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}}(2^{\epsilon n})$ implies that MKTP is hard for DET under logspace-uniform $\leq_{\mathrm{tt}}^{\mathsf{AC}^0}$ reductions (Theorem 15).

At this point, it is natural to ask if the circuit lower bounds are in fact *equivalent* to hardness of MKTP. We indeed show that this is the case, when we consider the minimum *oracle* circuit size problem. For an oracle A, MCSP^A is the set of pairs (T, s) such that T is computed by a size-s circuit that has "oracle gates" for A in addition to standard AND and OR gates. The related MKTP^A problem asks about the time-bounded Kolmogorov complexity of a string, when the universal Turing machine has access to the oracle A. For many oracles A that are hard for PH, we show that $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^A(2^{\epsilon n})$ for some $\epsilon > 0$ *if and only if* MKTP^A is hard for DET under a certain class of reducibilities (Theorem 17).

That is, it is impossible to prove hardness of MKTP^A (under some reducibilities) without proving circuit lower bounds, and vice versa. Our results clearly connect the fact that it is difficult to obtain hardness of MKTP^A with the fact that circuit size lower bounds are difficult.

Hardness under local reductions, and unconditional lower bounds. Murray and Williams [27] showed that MCSP and MKTP are not hard for TC^0 under so-called *local* reductions computable in time less than $\sqrt{n}$ – and thus in particular they are not hard under NC^0 reductions that are very uniform (i.e., there is no routine computable in time $t(n) < n^{.5-\epsilon}$ that, on input (n, i) outputs the $O(1)$ queries upon which the i-th output bit of such an NC^0 circuit depends). Murray and Williams speculated that this might be a promising first step toward showing that MCSP is not hard for NP under Dlogtime-uniform

AC^0 reductions, since it follows from [1] that any set that is hard for TC^0 under P-uniform AC^0 reductions is also hard for TC^0 under P-uniform NC^0 reductions. Indeed, the results of Murray and Williams led us to expect that MCSP and MKTP are not even hard for PARITY under non-uniform NC^0 reductions.

Contrary to these expectations, we show that MKTP is hard not only for TC^0 but even for the complexity class DET under non-uniform NC^0 reductions (Theorem 13). Consequently, MKTP is not in $\mathsf{AC}^0[p]$ for any prime p.[1] Note that it is still not known whether MCSP or $R_{\mathrm{KT}} = \{x : \mathrm{KT}(x) \geq |x|\}$ is in $\mathsf{AC}^0[p]$. It is known[2] [4] that neither of these problems is in AC^0. Under a plausible derandomization hypothesis, this non-uniform reduction can be converted into a logspace-uniform $\leq_{\mathrm{tt}}^{\mathsf{AC}^0}$ reduction that is an AND of NC^0-computable queries. Thus "local" reductions are more effective for reductions to MKTP than may have been suspected.

Implications among hardness conditions for MKTP and MCSP. No $\leq_{\mathrm{T}}^{\mathsf{P}}$ reductions are known between MKTP^A or MCSP^A for any A. Although most previous complexity results for one of the problems have applied immediately to the other, via essentially the same proof, there has not been any proven relationship among the problems. For the first time, we show that, for many oracles A, hardness for MCSP^A implies hardness for MKTP^A (Theorem 17).

A reduction that is not "oracle independent". Hirahara and Watanabe [20] observed that all of the then-known reductions to MCSP and MKTP were "oracle-independent", in the sense that, for any class $\mathcal{C}$ and reducibility $\leq_r$, all proofs that MCSP (or MKTP) is hard for $\mathcal{C}$ under $\leq_r$ also show that MCSP^A (MKTP^A) is also hard for $\mathcal{C}$. They showed that oracle-independent $\leq_{\mathrm{T}}^{\mathsf{P}}$-reductions cannot show hardness for any class larger than P.

This motivates the search for reductions that are *not* oracle-independent. We give a concrete example of a logspace-uniform $\leq_{\mathrm{ctt}}^{\mathsf{AC}^0}$ reduction that (under a plausible complexity assumption) reduces DET to MKTP. This is *not* an oracle independent reduction, since $\mathsf{MKTP}^{\mathsf{QBF}}$ is not hard for DET under this same class of reductions (Corollary 19).

A clearer picture of how hardness "evolves". It is instructive to contrast the evolution of the class of problems reducible to MKTP^A under different types of reductions, as A varies from very easy ($A = \emptyset$) to complex ($A = \mathsf{QBF}$). For this thought experiment, we assume the very plausible hypothesis that $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}(2^{\epsilon n})$. Restrictions of QBF give a useful parameterization for the complexity of A. Consider A varying from being complete for each level of PH (that is, quantified Boolean formulas with $O(1)$ alternations between $\forall$ and $\exists$ quantifiers), to instances of QBF with $\log^* n$ alternations, then to $O(\log n)$ alternations etc., through to $2^{\sqrt{\log n}}$ alternations, and until finally $A = \mathsf{QBF}$. Since $\mathsf{DSPACE}(n) \subseteq \mathsf{P}^A/\mathsf{poly}$, at some point in this evolution we have $\mathsf{DSPACE}(n) \subseteq \mathsf{io\text{-}SIZE}^A(2^{\epsilon n})$; it is plausible to assume that this doesn't happen until A has at least $\log n$ quantifier alternations, or more.

At all stages in this evolution $\mathsf{SZK} \subseteq \mathsf{BPP}^{\mathsf{MKTP}^A}$ [8], until at some point $\mathsf{BPP}^{\mathsf{MKTP}^A}$ expands to coincide with PSPACE [4]. Also, at all stages in this evolution $\mathsf{DET} \leq_{\mathrm{m}}^{\mathsf{NC}^0}$-reduces to MKTP^A (and even when $A = \mathsf{QBF}$ we do not know, for instance, if $\mathsf{NC}^3 \leq_{\mathrm{m}}^{\mathsf{NC}^0}$-reduces to MKTP^A). Thus these reductions behave "monotonically", in the sense that as the complexity

[1] Subsequent to our work, a stronger average-case lower bound against $\mathsf{AC}^0[p]$ was proved [19]. The techniques of [19] do not show how to reduce DET, or even smaller classes such as TC^0, to MKTP. Thus our work is incomparable to [19].

[2] Somewhat remarkably, Oliveira and Santhanam [29] have independently shown that MCSP and MKTP are hard for DET under non-uniform $\leq_{\mathrm{tt}}^{\mathsf{TC}^0}$ reductions. Their proof relies on self-reducibility properties of the determinant, whereas our proof relies on the fact that Graph Isomorphism is hard for DET [37]. Their results have the advantage that they apply to MCSP rather than merely to MKTP, but because it is not known whether $\mathsf{TC}^0 = \mathsf{P}$ they do not obtain unconditional lower bounds, as in Corollary 14.

of A increases, the class of problems reducible to MKTP^A does not shrink noticeably, and sometimes appears to grow markedly.

The situation is much more intriguing when we consider the *uniform* class of $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reductions that arise from derandomizing the nonuniform $\leq_{\mathrm{m}}^{\mathsf{NC}^0}$ reductions from DET. At the start, when $A = \emptyset$, we have DET reducing to MKTP^A, and this is maintained until A becomes complex enough so that $\mathsf{DSPACE}(n) \subseteq \mathsf{io\text{-}SIZE}^A(2^{\epsilon n})$. At this point, not only does DET not reduce to MKTP^A, but neither does PARITY! (See Theorem 17.)

This helps place the results of [9] in the proper context. In [9] strong evidence was presented against $\mathsf{MCSP}^{\mathsf{QBF}}$ being hard for, say, P under $\leq_{\mathrm{m}}^{\mathsf{L}}$ reductions, and this was taken as indirect evidence that MCSP itself should not be hard for P, since $\mathsf{MCSP} \in \mathsf{NP}$ and thus is much "easier" than the PSPACE-complete problem $\mathsf{MCSP}^{\mathsf{QBF}}$. However, we expect that MCSP^A and MKTP^A should behave somewhat similarly to each other, and it *can* happen that a class can reduce to MKTP (Theorem 15) and *not* reduce to MKTP^A for a more powerful oracle A (Corollary 19).

Hardness of the Gap problem. Our new hardness results for MKTP^A share with earlier reductions the property that they hold even for "Gap" versions of the problem. That is, for some $\epsilon > 0$, the reduction works correctly for any solution to the promise problem with "yes" instances $\{(x, s) : \mathrm{KT}^A(x) \leq s\}$ and "no" instances $\{(x, s) : \mathrm{KT}^A(x) > s + |x|^\epsilon\}$. However, we do not know if they carry over to instances with a wider "gap" between the Yes and No instances; earlier hardness results such as those of [4, 6, 8, 34] hold for a much wider gap (such as with the Yes instances having $\mathrm{KT}(x) < |x|^\epsilon$, and the no instances with $\mathrm{KT}(x) \geq |x|$), and this is one reason why they applied both to MKTP and to MCSP. Thus there is interest in whether it is possible to reduce MCSP with small "gap" to MCSP with large "gap". If this were possible, then MCSP and MKTP would be interreducible in some sense.

Earlier work [9] had presented unconditional results, showing that "gap" versions of MCSP could not be hard for TC^0 under $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$ reductions, unless those reductions had large "stretch" (mapping short inputs to long outputs). In Section 4, we show that BPP-Turing reductions among gap MCSP problems require large stretch, unless $\mathsf{MCSP} \in \mathsf{BPP}$.

Natural NP-intermediate Problems. In Section 3 we also consider gap MCSP problems where the "gap" is quite large (i.e., problems of approximating the minimum circuit size for a truth table of size n within a factor of $n^{1-o(1)}$). Problems of this sort are of interest, because of the role they play in the natural proofs framework of [32], if one is trying to prove circuit lower bounds of size $2^{o(n)}$. Our Theorem 6 shows that these problems are NP-intermediate in the sense that these do not lie in $\mathsf{P/poly}$ and are not NP-hard under $\mathsf{P/poly}$ reductions, under modest cryptographic assumptions (weaker than assuming that factoring or discrete log requires superpolynomial-size circuits, or assuming the existence of a one-way function). To the best of our knowledge, these problems are the first natural NP-intermediate problems under the existence of an arbitrary one-way function.

Our new insight on MCSP here is that, if the gap problems are NP-hard, then MCSP is "strongly downward self-reducible": that is, any instance of MCSP of size n can be reduced to instances of size n^ϵ. In the past, many natural problems have been shown to be strongly downward self-reducible (see [10]); Our contribution is to show that MCSP also has such a property (under the assumption that the gap MCSP problems are NP-hard). In fact, we also present a similar argument showing that a $n^{1-o(1)}$ approximation for CLIQUE is NP-intermediate if $\mathsf{NP} \not\subseteq \mathsf{P/poly}$.

2 Preliminaries

We assume the reader is familiar with standard DTIME and DSPACE classes. We also occasionally refer to classes defined by time-bounded *alternating* Turing machines: $\mathsf{ATIME}(t(n))$, or by simultaneously bounding time and the number of alternations between existential and universal configurations: $\mathsf{ATIME\text{-}ALT}(t(n), a(n))$.

We refer the reader to the text by Vollmer [40] for background and more complete definitions of the standard circuit complexity complexity classes

$$\mathsf{NC}^0 \subsetneq \mathsf{AC}^0 \subsetneq \mathsf{AC}^0[p] \subsetneq \mathsf{TC}^0 \subseteq \mathsf{NC}^1 \subseteq \mathsf{P/poly},$$

as well as the standard complexity classes $\mathsf{L} \subseteq \mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PH} \subseteq \mathsf{PSPACE}$. Between L and P in this list, there is one more class that plays an important role for us: DET is the class of problems that are reducible to the problem of computing the determinant of integer matrices, by NC^1-Turing reductions.

This brings us to the topic of reducibility. Let $\mathcal{C}$ be either a class of functions or a class of circuits. We say that $A \leq_{\mathrm{m}}^{\mathcal{C}} B$ if there is a function $f \in \mathcal{C}$ (or f computed by a circuit family in $\mathcal{C}$, respectively) such that $x \in A$ iff $f(x) \in B$. We will make use of $\leq_{\mathrm{m}}^{\mathsf{L}}, \leq_{\mathrm{m}}^{\mathsf{TC}^0}, \leq_{\mathrm{m}}^{\mathsf{AC}^0}$ and $\leq_{\mathrm{m}}^{\mathsf{NC}^0}$ reducibility. The more powerful notion of Turing reducibility also plays an important role in this work. Here, $\mathcal{C}$ is a complexity class that admits a characterization in terms of Turing machines or circuits, which can be augmented with an "oracle" mechanism, either by providing a "query tape" or "oracle gates". We say that $A \leq_{\mathrm{T}}^{\mathcal{C}} B$ if there is a oracle machine in $\mathcal{C}$ (or a family of oracle circuits in $\mathcal{C}$) accepting A, when given oracle B. We make use of $\leq_{\mathrm{T}}^{\mathsf{P/poly}}, \leq_{\mathrm{T}}^{\mathsf{BPP}}, \leq_{\mathrm{T}}^{\mathsf{P}}, \leq_{\mathrm{T}}^{\mathsf{L}}$ and $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reducibility; instead of writing $A \leq_{\mathrm{T}}^{\mathsf{P/poly}} B$ or $A \leq_{\mathrm{T}}^{\mathsf{BPP}} B$, we will more frequently write $A \in \mathsf{P}^B/\mathsf{poly}$ or $A \in \mathsf{BPP}^B$. Turing reductions that are "nonadaptive" – in the sense that the list of queries that are posed on input x does not depend on the answers provided by the oracle – are called *truth-table reductions*. We make use of $\leq_{\mathrm{tt}}^{\mathsf{AC}^0}$ and $\leq_{\mathrm{tt}}^{\mathsf{TC}^0}$ reducibility.

Kabanets and Cai [23] sparked renewed interest in MCSP and highlighted connections between MCSP and more recent progress in derandomization. They introduced a class of reductions to MCSP, which they called *natural reductions*. Recall that instances of MCSP are of the form (T, s) where s is a "size parameter". A $\leq_{\mathrm{m}}^{\mathsf{P}}$ reduction f is called *natural* if $f(x)$ is of the form $f(x) = (f_1(x), f_2(|x|))$. That is, the "size parameter" is the same, for all inputs x of the same length.

Whenever circuit families are discussed (either when defining complexity classes, or reducibilities), one needs to deal with the issue of *uniformity*. For example, the class AC^0 (corresponding to families $\{C_n : n \in \mathbb{N}\}$ of unbounded fan-in AND, OR, and NOT gates having size $n^{O(1)}$ and depth $O(1)$) comes in various flavors, depending on the complexity of computing the mapping $1^n \mapsto C_n$. When this is computable in polynomial time (or logarithmic space), then one obtains P-uniform AC^0 (logspace-uniform AC^0, respectively). If no restriction at all is imposed, then one obtains non-uniform AC^0. As discussed in [40], the more restrictive notion of Dlogtime-uniform AC^0 is frequently considered to be the "right" notion of uniformity to use when discussing small complexity classes such as $\mathsf{AC}^0, \mathsf{AC}^0[p]$ and TC^0. If these classes are mentioned with no explicit mention of uniformity, then Dlogtime-uniformity is intended. For uniform NC^1 the situation is somewhat more complicated, as discussed in [40]; there is wide agreement that the "correct" definition coincides with $\mathsf{ATIME}(O(\log n))$.

There are many ways to define time-bounded Kolmogorov complexity. The definition $\mathrm{KT}(x)$ was proposed in [4], and has the advantage that it is polynomially-related to circuit

size (when a string x is viewed as the truth-table of a function). $\mathrm{KT}(x)$ is the minimum, over all d and t, of $|d| + t$, such that the universal Turing machine U, on input (d, i, b) can determine in time t if the i-th bit of x is b. (More formal definitions can be found in [4].)

A *promise problem* consists of a pair of disjoint subsets (Y, N). A language A is a *solution* to the promise problem (Y, N) if $Y \subseteq A \subseteq \overline{N}$. A language B reduces to a promise problem via a type of reducibility $\leq_r$ if $B \leq_r A$ for *every* set A that is a solution to the promise problem.

3 GapMCSP

In this section, we consider the "gap" versions of MCSP and MKTP. We focus primarily on MCSP, and for simplicity of exposition we consider the "size" of a circuit to be the number of AND and OR gates of fan-in two. (NOT gates are "free"). The arguments can be adjusted to consider other circuit models and other reasonable measures of "size" as well. Given a truth-table T, let $\mathrm{CC}(T)$ be the size of the smallest circuit computing T, using this notion of "size".

▶ **Definition 1.** For any function $\epsilon\colon \mathbb{N} \to (0, 1)$, let Gap_ϵMCSP be the approximation problem that, given a truth-table T, asks for outputting a value $f(T) \in \mathbb{N}$ such that

$$\mathrm{CC}(T) \leq f(T) \leq |T|^{1-\epsilon(|T|)} \cdot \mathrm{CC}(T).$$

Note that this approximation problem can be formulated as the following promise problem. (See also [14] for similar comments.)

▶ Fact 2. Gap_ϵMCSP is polynomial-time Turing equivalent to the following promise problem (Y, N):

$$Y := \{\, (T, s) \mid \mathrm{CC}(T) < s/|T|^{1-\epsilon(|T|)} \,\},$$
$$N := \{\, (T, s) \mid \mathrm{CC}(T) > s + 1 \,\},$$

where T is a truth-table and $s \in \mathbb{N}$.

Proof. Given a solution A of Gap_ϵMCSP, one can compute an approximation $f(T)$ of $\mathrm{CC}(T)$ as follows:

$$f(T) := \min\{\, s \in \mathbb{N} \mid (T, s) \notin A \,\}$$

We claim that $f(T)$ satisfies the approximation guarantee as given in Definition 1. By the definition of $f(T)$, we have $(T, f(T)) \notin A$, which implies that $(T, f(T)) \notin Y$, and thus $\mathrm{CC}(T) \geq f(T)/|T|^{1-\epsilon(|T|)}$. Similarly, by the definition of $f(T)$, we have $(T, f(T) - 1) \in A$, which implies that $(T, f(T) - 1) \notin N$, and thus $\mathrm{CC}(T) \leq f(T)$. To summarize, we have $\mathrm{CC}(T) \leq f(T) \leq |T|^{1-\epsilon(|T|)} \cdot \mathrm{CC}(T)$ and thus $f(T)$ satisfies the Definition 1.

On the other hand, suppose that an approximation $f(T)$ of $\mathrm{CC}(T)$ is given. We can define a solution A of Gap_ϵMCSP as $A := \{\, (T, s) \mid f(T) < s \,\}$. We claim that A indeed satisfies the promise of Gap_ϵMCSP. If $(T, s) \in Y$, then $f(T) \leq |T|^{1-\epsilon(|T|)} \cdot \mathrm{CC}(T) < s$ and therefore $(T, s) \in A$. On the other hand, if $(T, s) \in N$, then $f(T) \geq \mathrm{CC}(T) > s + 1 \geq s$, which implies $(T, s) \notin A$. ◀

Note that Gap_ϵMCSP becomes easier when ϵ becomes smaller. If $\epsilon(n) = o(1)$, then (using the promise problem formulation) it is easy to see that Gap_ϵMCSP has a solution in $\mathsf{DTIME}(2^{n^{o(1)}})$, since the Yes instances have witnesses of length $|T|^{o(1)}$. However, it is

worth emphasizing that, even when $\epsilon(n) = o(1)$, $\mathrm{Gap}_\epsilon\mathsf{MCSP}$ is a canonical example of a combinatorial property that is useful in proving circuit size lower bounds of size $2^{o(n)}$, in the sense of [32]. Thus it is of interest that MCSP cannot reduce to $\mathrm{Gap}_\epsilon\mathsf{MCSP}$ in this regime under very general notions of reducibility, unless MCSP itself is easy.

▶ **Theorem 3.** *For any polynomial-time-computable nonincreasing $\epsilon(n) = o(1)$, if $\mathsf{MCSP} \in \mathsf{BPP}^{\mathrm{Gap}_\epsilon\mathsf{MCSP}}$ then $\mathsf{MCSP} \in \mathsf{BPP}$.*

A new idea is that the $\mathrm{Gap}_\epsilon\mathsf{MCSP}$ is "strongly downward self-reducible." We will show that any $\mathrm{Gap}_\epsilon\mathsf{MCSP}$ instance of length n is reducible to $n^{1-\epsilon}$ MCSP instances of length n^ϵ. To this end, we will exploit the following simple fact.

▶ **Lemma 4.** *For a function $f\colon \{0,1\}^n \to \{0,1\}$, a string $x \in \{0,1\}^k$ and $k \in \mathbb{N}$, let $f_x\colon \{0,1\}^{n-k} \to \{0,1\}$ be a function defined as $f_x(y) := f(x,y)$. Then, the following holds:*

$$\max_{x\in\{0,1\}^k} \mathrm{CC}(f_x) \le \mathrm{CC}(f) \le 2^k \cdot \left(\max_{x\in\{0,1\}^k} \mathrm{CC}(f_x) + 3 \right),$$

(In other words, $\max_{x\in\{0,1\}^k} \mathrm{CC}(f_x)$ gives an approximation of $\mathrm{CC}(f)$ within a factor of 2^k.)

Proof. We first claim that $\max_{x\in\{0,1\}^k} \mathrm{CC}(f_x) \le \mathrm{CC}(f)$. Indeed, let C be a minimum circuit that computes f and x be an arbitrary string of length k. For each $x \in \{0,1\}^k$, define a circuit C_x as $C_x(y) := C(x,y)$ on input $y \in \{0,1\}^{n-k}$. Then, since C_x computes f_x and the size of C_x is at most that of C, we have $\mathrm{CC}(f_x) \le \mathrm{CC}(f)$.

Next, we claim that $\mathrm{CC}(f) \le 2^k \cdot \big(\max_{x\in\{0,1\}^k} \mathrm{CC}(f_x) + O(1)\big)$. For any $x \in \{0,1\}^k$, let C_x be a minimum circuit that computes f_x. We build a circuit that computes $f =: f_\epsilon$ recursively as follows: $f_z(x,y) = (\neg x_1 \wedge f_{z0}(x_2,\dots,x_k,y)) \vee (x_1 \wedge f_{z1}(x_2,\dots,x_k,y))$ for any string z of length less than k, and $f_x(y) = C_x(y)$ for any $x \in \{0,1\}^k$. Since $\mathrm{CC}(f_z) \le \mathrm{CC}(f_{z0}) + \mathrm{CC}(f_{z1}) + 3$ we obtain

$$\begin{aligned} \mathrm{CC}(f) &\le \sum_{x\in\{0,1\}^k} C_x(y) + 3\cdot(2^k - 1) \\ &< 2^k \cdot \left(\max_{x\in\{0,1\}^k} \mathrm{CC}(f_x) + 3 \right). \end{aligned}$$

◀

Proof of Theorem 3. Let M be an oracle BPP Turing machine which reduces MCSP to $\mathrm{Gap}_\epsilon\mathsf{MCSP}$. Let $|T|^c$ be an upper bound for the running time of M, given a truth-table T, and let $|T| = 2^n$.

We recursively compute the circuit complexity of T by the following procedure: Run M on input T. If M makes a query S to the $\mathrm{Gap}_\epsilon\mathsf{MCSP}$ oracle, then divide S into consecutive substrings $S_1, \cdots, S_{2^k}$ of length $|S| \cdot 2^{-k}$ such that $S_1 \cdot S_2 \cdots S_{2^k} = S$ (where k is a parameter, chosen later, that depends on $|S|$), and compute the circuit complexity of each S_i recursively for each $i \in [2^k]$. Then continue the simulation of M, using the value $2^k \cdot \big(\max_{i\in[2^k]} \mathrm{CC}(S_i) + 3\big)$ as an approximation to $\mathrm{CC}(S)$.

We claim that the procedure above gives the correct answer. For simplicity, let us first assume that the machine M has zero error probability. It suffices to claim that the simulation of M is correct in the sense that every query of M is answered with a value that satisfies the approximation criteria of $\mathrm{Gap}_\epsilon\mathsf{MCSP}$. Suppose that M makes a query S. By the assumption on the running time of M, we have $|S| \le |T|^c = 2^{nc}$. By Lemma 4, we have

$$\mathrm{CC}(S) \le 2^k \cdot \left(\max_{i\in[2^k]} \mathrm{CC}(S_i) + 3 \right) \le 2^k \cdot (\mathrm{CC}(S) + 3).$$

In particular, the estimated value satisfies the promise of $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ if $2^k \cdot (\mathrm{CC}(S)+3) \leq |S|^{1-\epsilon(|S|)} \cdot \mathrm{CC}(S)$. Since we may assume without loss of generality that $\mathrm{CC}(S) \geq 3$, it suffices to make sure that $2^{k+1} \cdot \mathrm{CC}(S) \leq |S|^{1-\epsilon(|S|)} \cdot \mathrm{CC}(S)$. Let $|S| = 2^m$. Then, in order to satisfy $k+1 \leq (1-\epsilon(|S|)) \cdot m$, let us define $k := (1-\epsilon(|S|)) \cdot m - 1$. For this particular choice of k, the estimated value $2^k \cdot \left(\max_{i \in [2^k]} \mathrm{CC}(S_i) + 3\right)$ of the circuit complexity of S satisfies the promise of $\mathrm{Gap}_\epsilon\mathrm{MCSP}$, which implies that the reduction M computes the correct answer for MCSP.

Now we analyze the time complexity of the algorithm. Each recursive step makes at most 2^{2cn} many recursive calls, because there are potentially 2^{cn} many queries S of M, each of which may produce at most $2^k \leq 2^{cn}$ recursive calls. The length of each truth-table S_i that arises in one of the recursive calls is $|S_i| = |S| \cdot 2^{-k} = 2^{m-k} = 2^{\epsilon(|S|)\cdot m+1}$. We claim that $|S_i| \leq 2^{1+(n/2)}$ holds for sufficiently large n. Let us take n to be large enough so that $\epsilon(2^{n/2}) \leq 1/2c$. If $m \geq n/2$, then $|S_i| \leq 2^{\epsilon(2^m)\cdot m+1} \leq 2^{\epsilon(2^{n/2})\cdot cn+1} \leq 2^{1+(n/2)}$. Otherwise, since $m \leq n/2$ and $\epsilon(|S|) < 1$, we obtain $|S_i| \leq 2^{\epsilon(|S|)\cdot m+1} \leq 2^{1+(n/2)}$. Therefore, on inputs of length 2^n, each recursive call produces instances of length at most $2^{1+(n/2)}$. The overall time complexity can be estimated as $2^{c'n} \cdot 2^{c'n/2} \cdot 2^{c'n/4} \cdots = 2^{2c'n}$ for some constant c' (say, $c' = 3c$), which is a polynomial in the input length 2^n.

We note that the analysis above works even for *randomized* reductions that may err with exponentially small probability. Since we have proved that the algorithm runs in polynomial time, the probability that the algorithm makes an error is at most a polynomial times an exponentially small probability, which is still exponentially small probability (by the union bound). ◀

▶ Remark. If we drop the assumption that $\epsilon(n)$ be computable, then the proof of Theorem 3 still shows that if $\mathsf{MCSP} \in \mathsf{P}^{\mathrm{Gap}_\epsilon\mathrm{MCSP}}/\mathsf{poly}$ then $\mathsf{MCSP} \in \mathsf{P}/\mathsf{poly}$.

▶ **Corollary 5.** *Let $\epsilon(n) = o(1)$. If $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ has no solution in P/poly then $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ is not hard for NP (or even for MCSP) under $\leq_{\mathrm{T}}^{\mathsf{P}/\mathsf{poly}}$ reductions, and is thus NP-intermediate.*

Proof. This is immediate from the preceding remark. If $\mathsf{MCSP} \in \mathsf{P}^{\mathrm{Gap}_\epsilon\mathrm{MCSP}}/\mathsf{poly}$ then $\mathsf{MCSP} \in \mathsf{P}/\mathsf{poly}$, which in turn implies that $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ has a solution in P/poly. ◀

In what follows, we show that the assumption of Corollary 5 is true under very modest cryptographic assumptions. It is known that, for any constant $\epsilon > 0$, $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ is SZK-hard under $\leq_{\mathrm{T}}^{\mathsf{P}/\mathsf{poly}}$ reductions [8]. Here, we show that if SZK is not in P/poly, then for some $\epsilon(n) = o(1)$, $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ has no solution in P/poly. In fact, we can prove something *stronger*: If auxiliary-input one-way functions exist, then $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ is not in P/poly. We now describe auxiliary-input one-way functions.

Usually, the existence of cryptographically-secure one-way functions is considered to be essential for meaningful cryptography. That is, one requires a function f computed in polynomial time such that, for any algorithm A computed by polynomial-sized circuits, $\Pr_x[f(A(f(x))) = f(x)] = 1/n^{\omega(1)}$ where x is chosen uniformly at random from $\{0,1\}^n$. A weaker notion that has been studied in connection with SZK goes by the name *auxiliary-input one-way functions*. This is an indexed family of functions $f_y(x) = F(y,x)$, where $|x| = p(|y|)$ for some polynomial p, and F is computable in time polynomial in $|y|$, such that for some infinite set I, for any algorithm[3] A computed by polynomial-sized circuits, for all $y \in I$,

[3] We have chosen to define one-way functions in terms of security against non-uniform adversaries. It is also common to use the weaker notion of security against probabilistic polynomial-time adversaries, as in [38].

$\Pr_x[f_y(A(f_y(x))) = f_y(x)] = 1/n^{\omega(1)}$ where $n = |y|$ and x is chosen uniformly at random from $\{0,1\}^{p(n)}$. It is known that there are promise problems in SZK that have no solution in P/poly only if auxiliary-input one-way functions exist. (This is due to [31]; a good exposition can be found in [38, Theorems 7.1 & 7.5], based on earlier work of [30].)

▶ **Theorem 6.** *If auxiliary-input one-way functions exist, then there is a function $\epsilon(n) = o(1)$ such that $>$ Gap$_\epsilon$MCSP is* NP*-intermediate. $>$ (Namely,* Gap$_\epsilon$MCSP *has no solution in* P/poly *and $>$* Gap$_\epsilon$MCSP *is not* NP*-hard under $\leq_{\mathrm{T}}^{\mathsf{P/poly}}$ reductions.)*

▶ Remark. In particular, either one of the following implies that some Gap$_\epsilon$MCSP is NP-intermediate, since each implies the existence of auxiliary-input one-way functions:

1. the existence of cryptographically-secure one-way functions.
2. SZK is not in P/poly.

Proof. Let $F(y,x)$ define an auxiliary-input one-way family of functions $f_y(x)$ where $|x| = p(|y|)$ for some polynomial p. Let $S(n)$ be the size of the smallest circuit such that for some y of length n, $\Pr_x[f_y(A(f_y(x))) = f_y(x)] \geq 1/S(n)$ where $n = |y|$ and x is chosen uniformly at random from $\{0,1\}^{p(n)}$. By assumption $S(n)$ is not bounded by any polynomial. Let $e(n)$ be a nondecreasing unbounded function such that $n^{c_0 e(n^{c_0})} < S(n)$ for infinitely many n, where c_0 is a constant that we will pick later.

At this point, we make use of some standard derandomization tools, including the HILL pseudorandom generator [18], and pseudorandom function generators [15, 32]. First, we recall the HILL construction, phrased in terms of non-uniform adversaries:

▶ **Theorem 7** (see [18])**.** *Let $F(y,x)$ be computable uniformly in time polynomial in $|y|$, and let $\mu : \mathbb{N} \to [0,1]$. For any oracle L and any oracle circuit M of size $s(n)$, there is a size $(s(n)^{O(1)}/\mu(n^{O(1)}))$ circuit N such that the following holds for any n and y: If*

$$\left| \Pr_{|r|=2n}[M^L(y,r) = 1] - \Pr_{|x|=n}[M^L(y, G_{f_y}^{\mathrm{HILL}}(x)) = 1] \right| \geq \mu(n),$$

then

$$\Pr_{|x|=n}[F(y, N^L(y, F(y,x))) = F(y,x)] \geq \mu(n^{O(1)})/n^{O(1)},$$

where r and x are chosen uniformly at random. Here $G_{f_y}^{\mathrm{HILL}}$ is a pseudorandom generator, where $G_{f_y}^{\mathrm{HILL}}(x)$ is computable in time polynomial in $|y|$, as described in [18].

Theorem 7 states that if there exists a distinguisher with access to an oracle L that distinguishes the output of $G_{f_y}^{\mathrm{HILL}}$ from the uniform distribution, then oracle access to L suffices to invert f_y on a significant fraction of the inputs. We now argue that such a distinguisher can be computed by a circuit of size $n^{O(e(n))}$ with oracle gates for Gap$_{1/e(n)}$MCSP, where $e(n)$ is the slow-growing function that we defined earlier.

Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be a pseudorandom generator mapping strings of length n to strings of length $2n$, constructed from the generator $G_{f_y}^{\mathrm{HILL}}$. Furthermore, let $G_0(x)$ be the first n bits of $G(x)$, and let $G_1(x)$ be the second n bits of $G(x)$, so that $G(x) = G_0(x)G_1(x)$. We now make use of the pseudorandom function generator of Razborov and Rudich [32], with the following parameters.

For any string w of length k, let $G_w(x) : \{0,1\}^n \to \{0,1\}^n$ be defined as $G_w(x) = G_{w_1}(G_{w_2}(\ldots(G_{w_n}(x))\ldots))$. Define $G'_y(x,w)$ to be the first bit of $G_w(x)$, where here the subscript y refers back to the fact that G is defined from $G_{f_y}^{\mathrm{HILL}}$.

Now let $z(x)$ be the truth table of $G'_y(x,w)$ (viewed as a function of w). Since $|w| = k$, $|z(x)| = 2^k$. Since G'_y is computed in time polynomial in the length of x and w, $\mathrm{CC}(z(x)) < (n+k)^c$ for some constant c.

Now, let us choose k to be $(c+1)e(n)\log n$. It follows that $\mathrm{CC}(z(x)) < (n + (c+1)e(n)\log n)^c < n^{c+1} = (2^{(c+1)e(n)\log n})^{1/e(n)} = |z(x)|^{1/e(n)}$. That is, from the pseudorandom distribution, we always have $\mathrm{CC}(z(x)) < |z(x)|^{1/e(n)}$, whereas a random string has CC complexity at least roughly equal to the length of the string with high probability.

Thus, an oracle gate for any oracle that satisfies the $\mathrm{Gap}_{1/e(n)}$MCSP promise problem can distinguish random functions from the output of the pseudorandom function generator. And [32] shows how to obtain a circuit L of size $n^{O(e(n))}$ such that

$$\left| \Pr_{|r|=2n}[M^L(y,r)=1] - \Pr_{|x|=n}[M^L(y, G_{f_y}^{\mathrm{HILL}}(x))=1] \right| \geq 1/n^{e(n)}.$$

By Theorem 7, there are constants c_1, c_2, c_3, c_4 and there is a circuit N of size $n^{c_1 e(n^{c_2})}$ such that the following holds for any n and y:

$$\Pr_{|x|=n}[F(y, N^L(y, F(y,x))) = F(y,x)] \geq n^{c_3 e(n^{c_4})}$$

where x is chosen uniformly at random.

Now if we pick c_0 to be greater than $\max\{c_1, c_2, c_3, c_4\}$, it follows that N is a circuit of size less than $S(n)$ that inverts $f_y(x)$ with probability greater than $1/S(n)$, contrary to the definition of S.

This establishes that no solution to the $\mathrm{Gap}_{1/e(n)}$MCSP promise problem lies in P/poly. By Corollary 5, $\mathrm{Gap}_{1/e(n)}$MCSP is NP-intermediate. ◀

▶ Remark. Observe that Theorem 6 can also be rephrased in terms of *uniform* probabilistic adversaries, if we assume that the one-way functions require time $n^{e(n)}$ to invert, for some easy-to-compute function e.

3.1 Other NP-intermediate problems

Although our focus is primarily on MCSP, we observe here that the strongly downward self-reducibility property that we exploited above is fairly common. For instance, it has been noticed previously that CLIQUE also has this property [36, 10]. It appears to be a new observation, however, that this property yields a natural NP-intermediate optimization problem.

▶ **Definition 8.** For any function $\epsilon\colon \mathbb{N} \to (0,1)$, let Gap_ϵCLIQUE be the approximation problem that, given an n-vertex graph G, asks for outputting a value $f(G) \in \mathbb{N}$ such that

$$\omega(G) \leq f(G) \leq n^{1-\epsilon(n)} \cdot \omega(G).$$

Here, as usual $\omega(G)$ denotes the clique number of G: the size of the largest clique in G.

▶ **Theorem 9.** NP $\not\subseteq$ P/poly *if and only if there is an* $\epsilon(n) = o(1)$ *such that* Gap_ϵCLIQUE *has no solution in* P/poly *and is not hard for* NP *under* $\leq_{\mathrm{T}}^{\mathsf{P/poly}}$ *reductions.*

Proof. Assume NP $\not\subseteq$ P/poly. Define $e(n)$ to be the least c such that, for all $m \geq n$, there is a circuit of size m^c that computes a function $f(G)$ (for m-vertex graphs G) such that $\omega(G) \leq f(G) \leq m^{1-1/c} \cdot \omega(G)$. If $e(n) = O(1)$, it follows from [17] that CLIQUE $\in$ P/poly, contrary to assumption. Thus $e(n) = \omega(1)$.

Let $\epsilon = \epsilon(n) = 1/e(n)$; thus $\epsilon(n) = o(1)$. It follows immediately from the definition of $e(n)$ that Gap_ϵCLIQUE has no solution in P/poly.

If we partition the vertices of an n-node graph G into $n^{1-\epsilon}$ parts $V_1, \ldots, V_{n^{1-\epsilon}}$ of size at most $\lceil n^{\epsilon} \rceil$, then $\omega(G) \leq (n^{1-\epsilon}) \cdot \max_i \omega(G_i)$, where G_i is the induced subgraph of G with vertices in V_i. (See [36, 10] for other applications of this observation.)

Now, precisely as in the proof of Theorem 3, it follows that if CLIQUE were P/poly-Turing reducible to Gap_{ϵ}CLIQUE, then CLIQUE $\in$ P/poly, contrary to our assumption. This shows that Gap_{ϵ}CLIQUE is not NP-hard under P/poly reductions, and thus completes the "only if" direction of the Theorem. (The converse is trivial.) ◀

4 Reductions among GapMCSPs Require Large Stretch

In the previous section, we studied Gap_{ϵ}MCSP where $\epsilon(n) = o(1)$. In this section, we focus on the case where ϵ is a fixed positive constant.

In what follows, we say that a reduction from Gap_{δ}MCSP to Gap_{ϵ}MCSP *has stretch* n^c if, on input T, the reduction makes queries of length at most $|T|^c$.

▶ **Theorem 10.** *Let* $0 < \epsilon < \delta < 1$. *If* Gap_{δ}MCSP *is reducible to* Gap_{ϵ}MCSP *via a randomized Turing reduction of stretch at most* n^c *for some* $c < \delta/\epsilon$, *then* Gap_{δ}MCSP $\in$ BPP.

Proof. The argument is almost identical to the argument in the preceding section. Given an input to Gap_{δ}MCSP, simulate the reduction from Gap_{δ}MCSP to Gap_{ϵ}MCSP. As before, if the reduction makes a query S, then divide S into consecutive substrings $S_1, \ldots, S_{2^k}$ of length 2^{m-k}, where m is defined as $|S| = 2^m$ and k is a parameter chosen later depending on m. For each $i \in [2^k]$, recursively solve Gap_{δ}MCSP on the instance S_i, and let $f(S_i)$ be the answer of the recursive call. Now, we estimate the circuit complexity of S as $2^k \cdot (\max_{i \in [2^k]} f(S_i) + 3)$ and continue the simulation.

We claim the correctness of the simulation for a certain choice of parameter $k = k(m)$. Let e denote the estimated circuit complexity of S, that is, $e := 2^k \cdot (\max_{i \in [2^k]} f(S_i) + 3)$. The goal is to show that e satisfies the promise of Gap_{ϵ}MCSP, or equivalently,

$$\mathrm{CC}(S) \leq e \leq |S|^{1-\epsilon} \cdot \mathrm{CC}(S). \tag{1}$$

We may assume that answers of recursive calls satisfy the promise of Gap_{δ}MCSP by induction: that is, $\mathrm{CC}(S_i) \leq f(S_i) \leq |S_i|^{1-\delta} \cdot \mathrm{CC}(S_i)$. Thus, by Lemma 4, we have

$$e \geq 2^k \cdot \left(\max_{i \in [2^k]} \mathrm{CC}(S_i) + 3 \right) \geq \mathrm{CC}(S),$$

as required in the first inequality of (1). Now we turn to the second inequality of (1). We may assume, without loss of generality, that $e \leq 2^{k+1} \cdot \max_{i \in [2^k]} f(S_i)$. Therefore, we obtain

$$\begin{aligned}
e &\leq 2^{k+1} \cdot \max_{i \in [2^k]} f(S_i) \\
&\leq 2^{k+1} \cdot \max_{i \in [2^k]} |S_i|^{1-\delta} \cdot \mathrm{CC}(S_i) && \text{(by the promise of } \mathrm{Gap}_{\delta}\text{MCSP)} \\
&= 2^{k+1+(m-k)(1-\delta)} \cdot \max_{i \in [2^k]} \mathrm{CC}(S_i) && \text{(since } |S_i| = 2^{m-k}\text{)} \\
&\leq 2^{k+1+(m-k)(1-\delta)} \cdot \mathrm{CC}(S) && \text{(by Lemma 4)} \\
&\leq |S|^{1-\epsilon} \cdot \mathrm{CC}(S),
\end{aligned}$$

where the last inequality holds if $k+1+(m-k)(1-\delta) \leq m \cdot (1-\epsilon)$, that is, $k \leq m - m\epsilon/\delta - 1/\delta$. Thus we define k as $k := m - m\epsilon/\delta - 1/\delta$, which ensures the second inequality of (1).

Now we turn to analysis of the running time of the algorithm. Let 2^n be the length of the input to the algorithm. By the assumption on the stretch of the reduction, we have $|S| \leq 2^{nc}$, that is, $m \leq nc$. Therefore, $|S_i| = 2^{m-k} = 2^{mc\epsilon/\delta+1/\delta} \leq 2^{nc\epsilon/\delta+1/\delta}$. Since $c\epsilon/\delta < 1$, the algorithm above runs in polynomial time. (Indeed, let $t(N)$ be an upper bound of the running time of the algorithm on inputs of length N and $\rho := c\epsilon/\delta < 1$. We have $t(N) \leq N^{O(1)}t(N^\rho)$. Solving this recursive inequality, we obtain $t(N) = N^{O(1)}$.) ◀

5 Hardness for DET

In this section, we present some of our main contributions. We show that MKTP is hard for DET under $\leq_{\mathrm{m}}^{\mathsf{NC}^0}$ reductions (Theorem 13); prior to this, no variant of MCSP has been shown to be hard for any complexity class under any type of many-one reducibility. The $\leq_{\mathrm{m}}^{\mathsf{NC}^0}$ reduction that we present is nonuniform; we show that hardness under *uniform* reductions is closely related to lower bounds in circuit complexity, and in some cases we show that circuit lower bounds are *equivalent* to hardness results under uniform notions of reducibility (Theorem 17). These techniques allow us to prove the first results relating the complexity of MCSP^A and MKTP^A problems.

Here is the outline of this section. We will build on a randomized reduction of Allender, Grochow and Moore [5]: They showed that there is a ZPP reduction from the rigid[4] graph isomorphism problem to MKTP. Here we show that the reduction is in fact an AC^0 reduction (Corollary 12). Combining Torán's AC^0 reduction [37] from DET to the rigid graph isomorphism as well as the Gap theorem [2], we will show $\mathsf{DET} \leq_{\mathrm{m}}^{\mathsf{NC}^0} \mathsf{MKTP}$ (Theorem 13).

In order to establish equivalence between hardness under uniform AC^0 reductions and circuit size lower bounds, we will derandomize the reduction of [5] (Theorem 15). To this end, we first show that there is an AC^0 reduction f from the rigid graph isomorphism problem to MKTP and an "encoder" e that encodes random binary strings into a random permutation in Lemma 11 below.

▶ **Lemma 11.** *Let A be any oracle. There is a function f computable in Dlogtime-uniform AC^0 and a function e computable in Dlogtime-uniform TC^0 such that, for any two rigid graphs G, H with n vertices:*

- $\Pr_r[f(G, H, e(r)) \notin \mathsf{MKTP}^A] > 1 - \frac{1}{2^{4n^2}}$ *if $G \not\equiv H$, and*
- $\Pr_r[f(G, H, e(r)) \in \mathsf{MKTP}^A] = 1$ *if $G \equiv H$.*

Proof. We present the proof for $A = \emptyset$; however, it is immediate that the proof carries over for any oracle A. The function f is given by the reduction presented in [5, Theorem 1], showing that the Rigid Graph Isomorphism Problem is in $\mathsf{Promise\text{-}ZPP}^{\mathsf{MKTP}}$. This reduction takes graphs G and H as input, and interprets the random coin flip sequence r as a tuple (w, Π) where Π is a sequence of t random permutations $\pi_1, \ldots, \pi_t$, and $|w| = t$. To the best of our knowledge, there is not any encoding of permutations such that both

- a random string of polynomial length can be viewed as encoding an approximately uniformly random permutation, and
- an AC^0 function can efficiently decode the permutation and apply it to the adjacency matrix of a graph.

To get around this, we will employ a TC^0-computable mapping e from random strings to encodings of permutations. Such mappings have been used before (e.g., [3, 39]), but we provide a detailed exposition here for completeness.

[4] A graph is *rigid* if it has no nontrivial automorphisms.

First, note that a random string s of length $n^{\ell'}(\ell' \log n)$ divided into blocks of length $\ell' \log n$ will contain at least n blocks containing distinct entries, with probability $\geq 1 - 2^{-n^{\ell}}$ (where ℓ depends on ℓ').

Thus, with very high probability, a TC^0 routine can, on input s, output a sequence $((1, \sigma(1)), (2, \sigma(2)), \ldots, (n, \sigma(n)))$ with the property that $\sigma(i)$ is the contents of the i-th distinct block in s, and thus σ is an injective map $\sigma : [n] \to [n^{\ell'}]$. Now a second TC^0-computable transformation can map $\sigma(i)$ to the number $j = \pi(i)$ such that $\sigma(i)$ is the j-th element in a sorted list of the elements $\{\sigma(1), \ldots, \sigma(n)\}$. In this way, a random string s of length $n^{O(1)}$ yields a nearly-uniformly-random permutation $\pi = e(s)$, where π is encoded as a sequence of pairs $(i, \pi(i))$, which is an encoding that is amenable to manipulation by AC^0 circuits. (With probability at most $1/2^{n^{\ell}}$ the string s will not encode a permutation, in which case we define $e(s)$ to be a string of zeros.) Thus the TC^0 function $e(r)$ in the statement of the theorem takes approximately $t + tn^{\ell'}\ell' \log n$ bits and produces the string w and the t permutations, appropriately encoded. (We will pick an appropriate value for ℓ' later, based on the desired value of ℓ in the failure probability $1/2^{n^{\ell}}$.)

The reduction (as presented in [5]) then produces a string $x_r = \pi_1(G_{w_1}), \ldots, \pi_t(G_{w_t})$. The proof in [5] shows that, if $G \equiv H$, then $\mathrm{KT}(x_r) \leq t(\log n!) + t/3 + n^4$, whereas if $G \not\equiv H$ then $\mathrm{KT}(x_r) > C(w, \pi_1, \ldots, \pi_t) - (2n - 2)\log n$. It is immediate that

$$\Pr[C(w, \pi_1, \ldots, \pi_t) \leq t(\log n!) + t - 5n^2] \leq 1/2^{5n^2}$$

if the permutations are chosen uniformly at random. Thus, as in [5], we pick $t = n^5$ and choose the threshold $\theta = t(\log n!) + t/3 + n^4$. At this point we need to choose the constant ℓ'. Note that the probability that the decoding $e(r)$ of the random string r contains a string of zeros (indicating that one of the t blocks of r fails to decode to a permutation) is bounded by $t/2^{n^{\ell}}$, and note also that, as ℓ' increases, the induced distribution becomes ever closer to a uniform distribution on t independent permutations. Thus we choose ℓ' large enough so that $1/2^{5n^2} + n^5/2^{n^{\ell}}$ is enough less than $1/2^{4n^2}$. We then define $f(G, H, e(r)) = (x_r, \theta)$ (unless $e(r)$ contains a block of zeros, indicating failure, in which case $f(G, H, e(r))$ is a string of zeros). Now we observe that f is computable in AC^0. This is because the only real work performed by f involves permuting a graph. Graphs are encoded as adjacency matrices. Thus, given a graph G and a permutation π, the bit (r, s) $\pi(G)$ is the same as the bit (i, j) in G, where $\pi(i) = r$ and $\pi(j) = s$. That is, position (r, s) in the output is the $\mathrm{OR}_{i,j}$ (taken over all relevant positions (i, j) in the encoding of π) of [$G_{i,j}$ AND [the encoding of π contains the strings (i, r) and (j, s)]]. This latter condition can easily be checked in AC^0.

This establishes that f has the desired properties. (The argument presented in [5] uses a slightly different value of θ because they do not need the same error probability as we require here.) ◀

▶ **Corollary 12.** *Let A be any oracle. The rigid graph isomorphism problem is reducible to MKTP^A via a non-uniform $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$ reduction.*

Proof. A standard counting argument shows that there is a value of $e(r)$ that can be hardwired into the probabilistic reduction of Lemma 11 that works correctly for all pairs (G, H) of n-vertex graphs. (Note that the length of the input is $2n^2$, and the error probability is at most $1/2^{4n^2}$.) ◀

▶ **Theorem 13.** *Let A be any oracle.* DET *is reducible to MKTP^A via a non-uniform $\leq_{\mathrm{m}}^{\mathsf{NC}^0}$ reduction. Furthermore, this reduction is "natural" in the sense of [23].*

Proof. Since DET is closed under $\leq_m^{TC^0}$ reductions, it suffices to show that MKTP^A is hard under $\leq_m^{AC^0}$ reductions, and then appeal to the "Gap" theorem of [2], to obtain hardness under $\leq_m^{NC^0}$ reducibility. Torán [37] shows that DET is AC^0-reducible to GI. In fact it is shown in the proofs of Theorem 5.3 and Corollary 5.4 of [37] that DET is AC^0-reducible to GI via a reduction that produces only pairs of rigid graphs as output. Composing this reduction with the non-uniform AC^0 reduction given by Corollary 12 completes the argument. (Since DET is closed under complement, there is also a non-uniform $\leq_m^{AC^0}$ reduction to the complement of MKTP^A.)

Since the same threshold θ is used for all inputs of the same length, the reduction is "natural". ◀

An appeal to the circuit lower bounds of Razborov and Smolensky [33, 35] now yields the following corollary:

▶ **Corollary 14.** MKTP^A *is not in* $AC^0[p]$ *for any oracle* A *and any prime* p.

(An alternate proof of this circuit lower bound can be obtained by applying the pseudorandom generator of [13] that has sublinear stretch and is secure against $AC^0[p]$. Neither argument seems easy to extend, to provide a lower bound for MCSP.)

The reader may wonder whether the non-uniform reduction can be made uniform under a suitable derandomization hypothesis. We do not know how to obtain a uniform AC^0-many-one reduction, but we can come close, if the oracle A is not too complex. Recall the definition of ctt-reductions: $B \leq_{ctt}^{\mathcal{C}} C$ if there is a function $f \in \mathcal{C}$ with the property that $f(x)$ is a list $f(x) = (y_1, \ldots, y_m)$, and $x \in B$ if and only if $y_j \in C$ for all j. Furthermore, we say that f is a *natural* logspace-uniform $\leq_{ctt}^{AC^0}$-reduction to MKTP if each query y_j has the same length (and this length depends only on $|x|$), and furthermore each y_j is of the form (z_j, θ) where the threshold θ depends only on $|x|$.

The following theorem can be viewed as a "partial converse" to results of [27, 9], which say that problems in $\mathsf{LTH} \subseteq \mathsf{E}$ require exponential size circuits if MCSP or MKTP is hard for TC^0 under Dlogtime-uniform $\leq_m^{AC^0}$ reductions. That is, the earlier results show that very uniform hardness results imply circuit lower bounds, whereas the next theorem shows that somewhat stronger circuit lower bounds imply uniform hardness results (for a less-restrictive notion of uniformity, but hardness for a larger class). Later on, in Theorem 17, we present a related condition on reductions to MKTP^A that is *equivalent* to circuit lower bounds.

▶ **Theorem 15.** *Let* A *be any oracle. If there is some* $\epsilon > 0$ *such that* $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$*, then every language in* DET *reduces to* MKTP^A *via a natural logspace-uniform* $\leq_{ctt}^{AC^0}$*-reduction.*

Proof. Let $B \in \mathsf{DET}$. Thus there is an AC^0 reduction g reducing B to the Rigid Graph Isomorphism Problem [37]. Consider the following family of statistical tests $T_x(r)$, indexed by strings x:

On input r:
Compute $z = f(g(x), e(r))$, where $f(G, H, e(r))$ is the function from Lemma 11.
Accept iff ($x \in B$ iff $z \in \mathsf{MKTP}^A$).

Since $B \in \mathsf{DET} \subseteq \mathsf{P}$, the test $T_x(r)$ has a polynomial-size circuit with one MKTP^A oracle gate. (In fact, the statistical test is an NC^2 circuit with one oracle gate.) If $x \in B$, then T_x accepts *every* string r, whereas if $x \notin B$, T_x accepts most strings r.

Klivans and van Melkebeek [24] (building on the work of Impagliazzo and Wigderson [22]) show that, if $\mathsf{DSPACE}(n)$ requires exponential-size circuits from a given class $\mathcal{C}$, then

there is a hitting set generator computable in logspace that hits all large sets computable by circuits from $\mathcal{C}$ having size n^k. In particular, under the given assumption, there is a function h computable in logspace such that $h(0^n) = (r_1, r_2, \ldots, r_{n^c})$ with the property that, for all strings x of length n, there is an element of $h(0^n)$ that is accepted by T_x.

Now consider the logspace-uniform AC^0 oracle circuit family, where the circuit for inputs of length n has the strings $e(h(0^n)) = (e(r_1), e(r_2), \ldots, e(r_{n^c}))$ hardwired into it. The circuit computes the queries $f(g(x), e(r_i))$ for $1 \leq i \leq n^c$, and accepts if, for all i, $f(g(x), e(r_i)) \in \mathsf{MKTP}^A$. Note that if $x \notin B$, then one of the r_i is accepted by T_x, which means that $f(g(x), e(r_i)) \notin \mathsf{MKTP}^A$; if $x \in B$, then $f(g(x), e(r_i)) \in \mathsf{MKTP}^A$ for all i. This establishes that the reduction is correct. ◀

We remark that the hardness assumption $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$ can probably be weakened (saying that $\mathsf{DSPACE}(n)$ requires large circuits of some restricted sort), since the class of statistical tests that need to be fooled consists only of NC^2 circuits with one oracle gate. On the other hand, Theorem 17 indicates that the hardness assumption that we use is *equivalent* to the existence of uniform reductions, for certain oracles A – so it is not clear that there is much to be gained by searching for a weaker hardness assumption.

It is also possible to *strengthen* the hardness assumption, to obtain a stronger conclusion regarding the uniformity condition:

▶ **Corollary 16.** *Let A be any oracle. If there is some $\epsilon > 0$ such that the linear-time counting hierarchy* LCH *(see [11] for a definition) is not contained in* $\mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$*, then every language in* DET *reduces to* MKTP^A *via a natural (logspace-uniform-*TC^0*)-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction.*

Since (logspace-uniform-TC^0)-uniform AC^0 is somewhat cumbersome to work with, we concentrate on more familiar uniformity conditions such as (Dlogtime, logspace, P)-uniformity in the rest of the paper (although we observe here that Theorem 17 can also be rephrased in terms of the circuit condition of Corollary 16.

Theorem 15 deals with the oracle problem MKTP^A, but the most interesting case is the case where $A = \emptyset$, both because the hypothesis seems most plausible in that case, and because MKTP has been studied in connection with MCSP, which has been studied more than the associated oracle circuit problem MCSP^A. The hypothesis is false when $A = \mathsf{QBF}$, since the KT^A measure is essentially the same as the KS measure studied in [4], where it is shown that $\mathsf{PSPACE} = \mathsf{ZPP}^{R_{\mathrm{KS}}}$, and thus has polynomial-size $\mathsf{MKTP}^{\mathsf{QBF}}$-circuits. Strikingly, it is of interest that not only the hypothesis is false in this case – but the conclusion is false as well. (See Corollary 19.)

For certain oracles (and we discuss below how broad this class of oracles is), the existence of uniform reductions is *equivalent to* certain circuit lower bounds.

▶ **Theorem 17.** *Let* $\mathsf{MKTP}^A \in \mathsf{P}^A/\mathsf{poly}$*. Then the following are equivalent:*

- PARITY *reduces to* MKTP^A *via a natural logspace-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction.*
- *For some* $\epsilon > 0$, $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^A(2^{\epsilon n})$.
- *For some* $\epsilon > 0$, $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$.
- DET *reduces to* MKTP^A *via a natural logspace-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction.*

Furthermore, if PARITY *reduces to* MCSP^A *via a natural logspace-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction, then all of the above hold.*

Proof. First, we show that the first condition implies the second.

Let $\{C_n : n \in \mathbb{N}\}$ be a logspace-uniform family of oracle circuits computing PARITY, consisting of AC^0 circuitry feeding into oracle gates, which in turn are connected to an AND

gate as the output gate. Let the oracle gates in C_n be $g_1, g_2, \ldots, g_{n^c}$. On any input string x, let the value fed into gate g_i on input x be $(q_i(x), \theta)$, and recall that, since the reduction is natural, the threshold θ depends only on n, and thus it is a constant in C_n.

Now, we appeal to [9, Claim 3.11], and conclude that each $\mathsf{MKTP}^{\mathsf{QBF}}$ oracle gate can be replaced by a DNF formula of size at most $n^{O(1)}2^{O(\theta^2 \log \theta)}$. Inserting these DNF formulae into C_n (in place of each oracle gate) results in a circuit of size $n^{O(1)}2^{O(\theta^2 \log \theta)}$ computing PARITY. Let the depth of this circuit be some constant d. It follows from [16] that $n^{O(1)}2^{O(\theta^2 \log \theta)} \geq 2^{\Omega(n^{1/(d-1)})}$, and hence that $\theta \geq n^{1/4d}$.

Note that all of the oracle gates g_i must output 1 on input $0^{n-1}1$, and one of the oracle gates g_{i_0} must output 0 on input 0^n. Thus we have $\mathrm{KT}^A(q_{i_0}(0^n)) \geq \theta \geq n^{1/4d}$. It follows from [4, Theorem 11] that the function with truth-table $q_{i_0}(0^n)$ has no circuit (with oracle gates for A) of size less than $(\mathrm{KT}^A(q_{i_0}(0^n)))^{1/5} \geq \theta \geq n^{1/20d}$.

Note that, in order to compute the j-th bit of some query $q_i(0^n)$, it suffices to evaluate a logspace-uniform AC^0 circuit where all of the input bits are 0. Since this computation can be done in logspace on input $(0^n 1^i 0^j)$, note that the language $H = \{(n, i, j) : \text{the } j\text{-th bit of query } q_i(0^n) \text{ is } 1\}$ is in linear space. Let $m = |(n, i, j)|$, and let $s(m)$ be the size of the smallest circuit D_m computing H for inputs of length m. Hardwire the bits for n and also set the bits for i to i_0. The resulting circuit on $|j| < m$ bits computes the function given by $q_{i_0}(0^n)$, and it was observed above that this circuit has size at least $n^{1/20d} \geq 2^{m/20d}$.

This establishes the first implication. (Note also that a similar argument yields the same conclusion from the assumption that PARITY reduces to MCSP^A via a natural logspace-uniform $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$-reduction.)

The assumption that $\mathsf{MKTP}^A \in \mathsf{P}^A/\mathsf{poly}$ suffices to show that the second condition implies the third. More formally, we'll consider the contrapositive. Assume that $\mathsf{DSPACE}(n) \subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$ for every $\epsilon > 0$. An oracle gate for MKTP^A on inputs of size m can be replaced by a circuit (with oracle gates for A) of size m^c for some constant c. Carrying out this substitution in a circuit (with oracle gates for MKTP^A) of size $2^{\epsilon n}$ yields a circuit of size at most $2^{\epsilon n} + 2^{\epsilon n}(2^{\epsilon n})^c$.

Let $\delta > 0$. Then we can pick ϵ small enough so that $2^{\epsilon n} + 2^{\epsilon n}(2^{\epsilon n})^c < 2^{\delta n}$, thereby establishing that $\mathsf{DSPACE}(n) \subseteq \mathsf{io\text{-}SIZE}^A(2^{\delta n})$ for every $\delta > 0$. This establishes the second implication.

Theorem 15 establishes that the third condition implies the fourth. The fourth condition obviously implies the first. ◀

To the best of our knowledge, this is the first theorem that has given conditions where the existence of a reduction to MCSP^A implies the existence of a reduction to MKTP^A. We know of no instance where the implication goes in the opposite direction.

The logspace uniformity condition in Theorem 17 can be replaced by other less-restrictive uniformity conditions. We mention the following example:

▶ **Corollary 18.** *Let* $\mathsf{MKTP}^A \in \mathsf{P}^A/\mathsf{poly}$. *Then the following are equivalent:*

- PARITY *reduces to* MKTP^A *via a natural* P*-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction.*
- *For some* $\epsilon > 0$, $\mathsf{E} \not\subseteq \mathsf{io\text{-}SIZE}^A(2^{\epsilon n})$.
- *For some* $\epsilon > 0$, $\mathsf{E} \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$.
- DET *reduces to* MKTP^A *via a natural* P*-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction.*

Furthermore, if PARITY *reduces to* MCSP^A *via a natural* P*-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$*-reduction, then all of the above hold.*

At this point, we should consider the class of oracles for which Theorem 17 applies. That is, what is the set of oracles A for which $\mathsf{MKTP}^A \in \mathsf{P}^A/\mathsf{poly}$? First, we observe that this condition holds for any PSPACE-complete set, which yields the following corollary:

▶ **Corollary 19.** PARITY *does not reduce to either* $\mathsf{MKTP}^{\mathsf{QBF}}$ *or* $\mathsf{MCSP}^{\mathsf{QBF}}$ *via a natural logspace-uniform* $\leq^{\mathsf{AC}^0}_{\mathsf{ctt}}$*-reduction.*

As another example of a set for which $\mathsf{MKTP}^A \in \mathsf{P}^A/\mathsf{poly}$, consider the set $A = \{(M, x, 1^m) : M$ is an alternating Turing machine that accepts x, and runs in time at most m and makes at most $\log m$ alternations$\}$. A is complete for the class $\mathsf{ATIME\text{-}ALT}(n^{O(1)}, O(\log n))$ under $\leq^{\mathsf{AC}^0}_{\mathrm{m}}$ reductions. It is easy to see that $\mathsf{MKTP}^A \in \mathsf{ATIME\text{-}ALT}(n^{O(1)}, O(\log n))$, and thus $\mathsf{MKTP}^A \in \mathsf{P}^A$. (Other examples can easily be created in this way, using an even smaller number of alternations. Note that, for this oracle A, it seems plausible that all four conditions in Theorem 17 hold.

Nonetheless, we do grant that this does seem to be a strong condition to place upon the oracle A – and it has even stronger consequences than are listed in Theorem 17. For instance, note that the proof that the first condition in Theorem 17 implies the second relies only on the fact that PARITY requires large AC^0 circuits. Thus, an identical proof shows that these four conditions are also equivalent to the condition that PARITY is reducible to MKTP^A via a natural ctt-reduction where the queries are computed by logspace-uniform $\mathsf{AC}^0[7]$ circuits. (Or you can substitute any other problem and class of mod circuits, where an exponential lower bound is known because of [33, 35].) In fact, as in [9, Lemma 3.10] we can apply random restrictions in a logspace-uniform way (as described in [1]) and obtain a reduction from PARITY to MKTP^A where the queries are computed by logspace-uniform NC^0 circuits! That is, for example, MAJORITY is reducible to MKTP^A via reductions of this sort computed by logspace-uniform $\mathsf{AC}^0[3]$ circuits iff PARITY is reducible to the same set via reductions where the queries are computed by logspace-uniform NC^0 circuits. We find these implications to be surprising. The "gap" phenomenon that was described in [2] (showing that completeness under one class of reductions is equivalent to completeness under a more restrictive class of reductions) had not previously been observed to apply to $\mathsf{AC}^0[p]$ reducibility.

We want to highlight some contrasts between Theorem 15 and Corollary 19. $\mathsf{MKTP}^{\mathsf{QBF}}$ is hard for PSPACE under ZPP-Turing reductions [4], whereas MKTP is in NP. Thus $\mathsf{MKTP}^{\mathsf{QBF}}$ appears to be much harder than MKTP. Yet, under a plausible hypothesis, MKTP is hard for a well-studied subclass of P under a type of reducibility, where the "harder" problem $\mathsf{MKTP}^{\mathsf{QBF}}$ cannot even be used as an oracle for PARITY under this same reducibility.

In other words, the (conditional) natural logspace-uniform $\leq^{\mathsf{AC}^0}_{\mathsf{ctt}}$ reductions from problems in DET to MKTP given in Theorem 15 are not "oracle independent" in the sense of [20]. Prior to this work, there had been no reduction to MCSP or MKTP that did not work for every MCSP^A or MKTP^A, respectively.

Prior to this work, it appears that there was no evidence for any variant of MCSP or MKTP being hard for a reasonable complexity class under $\leq^{\mathsf{L}}_{\mathrm{T}}$ reductions. All prior reductions (such as those in [8, 4, 5]) had been probabilistic and/or non-uniform, or (even under derandomization hypotheses) seemed difficult to implement in NC. We had viewed the results of [9] as providing evidence that none of these variants would be hard for P under, say, logspace reducibility. Now, we are no longer sure what to expect.

Surprisingly (to us), the notion of uniformity appears to be central. In particular, the reader is probably wondering whether the logspace-uniformity condition in Theorem 15 can be improved to Dlogtime-uniformity. One answer is provided by the results of [12], which

show that – at a minimum – any such proof would need to involve a different approach than an implementation of the Nisan-Wigderson generator [28], because no set system that satisfies the Nisan-Wigderson requirements can be implemented in Dlogtime-uniform AC^0. However, Theorem 21 below shows that, under a plausible hypothesis, *no* Dlogtime-uniform approach is possible (at least, for many oracles A).

First, we recall Corollary 3.7 of [9], which states that $\mathsf{MKTP}^{\mathsf{QBF}}$ is not hard for P under $\leq^{\mathsf{L}}_{\mathrm{m}}$ reductions unless $\mathsf{PSPACE} = \mathsf{EXP}$. It turns out that this holds even for logspace-Turing reductions.

▶ **Theorem 20.** $\mathsf{MKTP}^{\mathsf{QBF}}$ *is not hard for* P *(or* NP*) under* $\leq^{\mathsf{L}}_{\mathrm{T}}$ *reductions unless* $\mathsf{PSPACE} = \mathsf{EXP}$ *(*$\mathsf{PSPACE} = \mathsf{NEXP}$*, respectively).* $\mathsf{MKTP}^{\mathsf{QBF}}$ *is not hard for* PSPACE *under* $\leq^{\mathsf{L}}_{\mathrm{T}}$ *reductions. The same holds for* $\mathsf{MCSP}^{\mathsf{QBF}}$*.*

We include this proof here, both because it improves a Corollary in [9], and because the proof can be viewed as a warm-up for the proof of Theorem 21.

Proof. First, note that $\leq^{\mathsf{L}}_{\mathrm{T}}$ and $\leq^{\mathsf{L}}_{\mathrm{tt}}$ reducibilities coincide [26]. Thus assume that $\mathsf{MKTP}^{\mathsf{QBF}}$ is hard for P under $\leq^{\mathsf{L}}_{\mathrm{tt}}$ reductions; we will show that $\mathsf{PSPACE} = \mathsf{EXP}$. (The proof for $\mathsf{MCSP}^{\mathsf{QBF}}$ is identical, and the variant concerning hardness for NP is analogous.)

The proof idea is based on [20]: Assume that $\mathsf{P} \subseteq \mathsf{L}^{\mathrm{MKTP}^{\mathrm{QBF}}}_{\mathrm{tt}}$. (Here, L_{tt} means a $\leq^{\mathsf{L}}_{\mathrm{tt}}$ reduction.) By standard padding, we obtain $\mathsf{EXP} \subseteq \mathsf{PSPACE}^{\mathrm{MKTP}^{\mathrm{QBF}}}_{\mathrm{tt}}$. Any query of a $\mathsf{PSPACE}_{\mathrm{tt}}$ machine has low $\mathrm{KT}^{\mathrm{QBF}}$ complexity. Moreover, one can check whether a string has low $\mathrm{KT}^{\mathrm{QBF}}$ complexity in PSPACE. Combining these two facts, we obtain $\mathsf{EXP} \subseteq \mathsf{PSPACE}^{\mathrm{MKTP}^{\mathrm{QBF}}}_{\mathrm{tt}} = \mathsf{PSPACE}$. A formal proof follows.

Let $B \in \mathsf{EXP}$. Let $B' = \{x10^{2^{|x|}} : x \in B\}$ and note that $B' \in \mathsf{P}$. Consider the $\leq^{\mathsf{L}}_{\mathrm{tt}}$ reduction that reduces B' to MKTP^A. On any input string y, let the i-th oracle query be $q_i(y)$. The language $\{(i,j,x) :$ the j-th input bit of $q_i(x10^{2^{|x|}})$ is $1\}$ is in PSPACE and thus is in $\mathsf{P}^{\mathsf{QBF}}$. It follows that $q_i(x10^{2^{|x|}})$ is of the form (y_i, θ_i), where $\mathrm{KT}^{\mathsf{QBF}}(y_i) = |x,i,j|^{O(1)}$. Thus, to evaluate the oracle query q_i on input $x10^{2^{|x|}}$, this PSPACE computation (on input x) suffices: Compute the bits of θ_i; this can be done in PSPACE, since the number of bits in θ_i is at most $|x|^{O(1)}$, and each bit is computable in PSPACE. If $\theta_i > |x,i,j|^c$ (for the appropriate value of c), then return "1" since the query y_i certainly has KT^A complexity less than this. Otherwise, try all descriptions d of length at most θ_i, to determine whether there is some such d for which $U^{\mathsf{QBF}}(d,j)$ is equal to the j-th bit of q_i (allowing at most $|x,i,j|^c$ steps for the computation of U).

The rest of the $\leq^{\mathsf{L}}_{\mathrm{tt}}$ reduction on input $x10^{2^{|x|}}$ can be computed in space $|x|^{O(1)}$, by re-computing the values of the oracle queries, as required.

The unconditional result that $\mathsf{MKTP}^{\mathsf{QBF}}$ is not hard for PSPACE under $\leq^{\mathsf{L}}_{\mathrm{T}}$ reductions follows along the same lines, choosing $B \in \mathsf{EXPSPACE}$, and leading to the contradiction $\mathsf{EXPSPACE} = \mathsf{PSPACE}$. ◀

A similar approach yields the following result:

▶ **Theorem 21.** *Let* $\mathsf{NP} \subseteq \mathsf{P}^A/\mathsf{poly}$*. If* $\mathsf{PSPACE} \not\subseteq \mathsf{PH}^A$*, then neither* MKTP^A *nor* MCSP^A *is hard for* NC^1 *under Dlogtime-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{tt}}$ *reductions.*

Proof. We present the proof for MKTP^A; the proof for MCSP^A is identical.

Assume that MKTP^A is hard for NC^1 under Dlogtime-uniform $\leq^{\mathsf{AC}^0}_{\mathrm{tt}}$; we will show that $\mathsf{PSPACE} \subseteq \mathsf{PH}^A$. Since we are assuming that $\mathsf{NP} \subseteq \mathsf{P}^A/\mathsf{poly}$ note that we also have $\mathsf{PH} \subseteq \mathsf{P}^A/\mathsf{poly}$.

By the closure properties of PH, it will suffice to show that $\mathsf{ATIME}(n) \subseteq \mathsf{PH}^A$.

Let $B \in \mathsf{ATIME}(n)$. Let $B' = \{x10^{2^{|x|}} : x \in B\}$ and note that $B' \in \mathsf{NC}^1$. Consider the oracle family (C_m) that reduces B' to MKTP^A. Let the oracle gates in C_{2^n+n+1} be $g_1, g_2, \ldots, g_\ell$. On any input string y, let the query that is fed into gate g_i be $q_i(y)$. The language $\{(2^{|x|} + |x| + 1, i, j, x) :$ the j-th input bit of $q_i(x10^{2^{|x|}})$ is $1\}$ is in PH and thus is in $\mathsf{P}^A/\mathsf{poly}$. It follows that $q_i(x10^{2^{|x|}})$ is of the form (y_i, θ_i), where $\mathrm{KT}^A(y_i) = |x, i, j|^{O(1)}$. Thus, to evaluate oracle gate g_i on input $x10^{2^{|x|}}$, this PH^A computation (on input x) suffices: Compute the bits of θ_i; this can be done in PH, since the number of bits in θ_i is at most $|x|^{O(1)}$, and each bit is computable in PH. If $\theta_i > |x, i, j|^c$ (for the appropriate value of c), then return "1" since the query y_i certainly has KT^A complexity less than this. Otherwise, guess a description d of length at most θ_i, and universally check (for each j) that $U^A(d, j)$ is equal to the j-th bit of q_i (allowing at most $|x, i, j|^c$ steps for the computation of U).

To evaluate the rest of the circuit, note that the unbounded fan-in AND and OR gates that sit just above the oracle gates can also be evaluated in PH^A (at one level higher in the hierarchy than is required to evaluate the oracle gates). Repeating this process through the remaining $O(1)$ levels of the circuit yields the desired PH^A algorithm for B. ◀

▶ Remark. The significance of Theorem 21 is best viewed by combining it with Theorem 15. If we choose A to be any NP-complete set, then both of the hypotheses $\mathsf{PSPACE} \not\subseteq \mathsf{PH}^A (= \mathsf{PH})$ and $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$ are plausible. Thus, for such oracles A, under a plausible hypothesis, we have both MKTP^A is *not* hard for NC^1 under Dlogtime-uniform $\leq^{\mathsf{AC}^0}_{\mathrm{tt}}$ reductions, and MKTP^A *is* hard for DET under logspace-uniform $\leq^{\mathsf{AC}^0}_{\mathrm{ctt}}$ reductions. Thus different notions of uniformity are a key part of the puzzle, when trying to understand the hardness of problems such as MKTP and MCSP.

We are even able to extend our approach in some cases, to apply to AC^0-Turing reducibility.

▶ **Theorem 22.** *Let* $\mathsf{NP}^A \subseteq \mathsf{P}^A/\mathsf{poly}$. *If* $\mathsf{PSPACE} \not\subseteq \mathsf{PH}^A$, *then neither* MKTP^A *nor* MCSP^A *is hard for* NC^1 *under Dlogtime-uniform* $\leq^{\mathsf{AC}^0}_{\mathrm{T}}$ *reductions.*

Proof. Note that in a circuit computing an $\leq^{\mathsf{AC}^0}_{\mathrm{T}}$ reduction, there is an "initial" layer of oracle gates, whose queries are computed nonadaptively, while all oracle gates at deeper levels have queries whose values depend upon oracle gates at earlier levels in the circuit. Note also that, under the given assumption $\mathsf{NP}^A \subseteq \mathsf{P}^A/\mathsf{poly}$, we can conclude that $\mathsf{PH}^A \subseteq \mathsf{P}^A/\mathsf{poly}$.

The proof now proceeds along precisely the same lines as the proof of Theorem 21, which shows that a PH^A computation can compute the value of each wire that feeds into the "initial" layer of oracle gates. Similarly, as in the proof of Theorem 21, all of the AND, OR, and NOT gates at higher levels can be computed in PH^A, given that the gates at lower levels can be evaluated in PH^A. Thus, we need only show how to deal with oracle gates at deeper levels.

Consider any such oracle gate g. On any input string y, let the query that is fed into gate g when evaluating the circuit on input y be $q_g(y)$. The language $\{(2^{|x|} + |x| + 1, g, j, x)$: the j-th input bit of $q_g(x10^{2^{|x|}})$ is $1\}$ is in PH^A and thus (by our new assumption) is in $\mathsf{P}^A/\mathsf{poly}$. It follows that $q_g(x10^{2^{|x|}})$ is of the form (y, θ), where $\mathrm{KT}^A(y) = |x, g, j|^{O(1)}$. Thus, to evaluate oracle gate g on input $x10^{2^{|x|}}$, this PH^A computation (on input x) suffices: Compute the bits of θ; this can be done in PH^A, since the number of bits in θ_i is at most $|x|^{O(1)}$, and each bit is computable in PH. If $\theta_i > |x, g, j|^c$ (for the appropriate value of c), then return "1" since the query y certainly has KT^A complexity less than this. Otherwise, guess a description d of length at most θ, and universally check (for each j) that $U^A(d, j)$ is equal to the j-th bit of q_g (allowing at most $|x, i, j|^c$ steps for the computation of U). ◀

A consequence of Theorem 22 is the following corollary, which has the same flavor of results of the form "MCSP is hard for class $\mathcal{C}$ implies a likely but hard-to-prove consequence" as presented by Murray and Williams [27], but moving beyond the $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$ reductions considered by them, to the more general $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reductions.

▶ **Corollary 23.** *If either of* MKTP *or* MCSP *is hard for* NC^1 *under Dlogtime-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions, then* $\mathsf{NP} \neq \mathsf{NC}$.

Proof. This follows from Theorem 21 when $A = \emptyset$. If $\mathsf{NP} = \mathsf{NC}$, then we have $\mathsf{NP} \subseteq \mathsf{P/poly}$, and $\mathsf{PH} = \mathsf{NC} \neq \mathsf{PSPACE}$. Thus neither MKTP nor MCSP is hard for NC^1 under Dlogtime-uniform $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reductions. ◀

We also present another result in this vein, about NP-completeness. Prior work [27, 9] had obtained stronger consequences from the stronger assumption that MCSP is NP-complete under Dlogtime-uniform $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$ reductions.

▶ **Corollary 24.** *If either of* MKTP *or* MCSP *is hard for* NP *under Dlogtime-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions, then*

$$\mathsf{NP} \neq \mathsf{MA} \cap \mathsf{P/poly}.$$

Proof. If you modify the proof of Theorem 22, replacing NC^1 by NP and replacing PSPACE by NEXP, you obtain that, if $\mathsf{NP} \subseteq \mathsf{P/poly}$, then $\mathsf{NEXP} \neq \mathsf{PH}$ implies that neither MKTP nor MCSP is hard for NP under Dlogtime-uniform $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reductions.

Or, restating this using the same hypothesis as in the statement of the corollary, if MKTP or MCSP is hard for NP under Dlogtime-uniform $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$, then either $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ or $\mathsf{NEXP} = \mathsf{PH}$. Since ($\mathsf{NP} \subseteq \mathsf{P/poly}$ and $\mathsf{NEXP} = \mathsf{PH}$) is equivalent to $\mathsf{NEXP} \subseteq \mathsf{P/poly}$, and since $\mathsf{NEXP} \subseteq \mathsf{P/poly}$ is equivalent to $\mathsf{NEXP} = \mathsf{MA}$ [21], we obtain that NP-hardness of MCSP or MKTP implies $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ or $\mathsf{NEXP} = \mathsf{MA}$. (Murray and Williams obtain essentially this same consequence under the stronger assumption that MCSP is complete under $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$ reductions, but are also able to show that $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$ in this case.)

In either case, we obtain the consequence $\mathsf{NP} \neq \mathsf{MA} \cap \mathsf{P/poly}$. ◀

We close this section with another variant of Theorem 22, proved via the same technique:

▶ **Theorem 25.** *Let* $\mathsf{NP}^A \subseteq \mathsf{P}^A/\mathsf{poly}$. *If* $\mathsf{NEXP} \not\subseteq \mathsf{PSPACE}^A$ *(or* $\mathsf{NEXP} \not\subseteq \mathsf{EXP}^A$*), then neither* MKTP^A *nor* MCSP^A *is hard for* NP *under logspace-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions (*P*-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions, respectively).*

▶ **Corollary 26.** $\mathsf{MKTP}^{\mathsf{QBF}}$ *is not hard for* NP *under logspace-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions (*P*-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions) unless* PSPACE = NEXP *(*EXP = NEXP*, respectively). The same holds for* $\mathsf{MCSP}^{\mathsf{QBF}}$.

Although the following corollary discusses $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reductions, it also says something about $\leq_{\mathrm{T}}^{\mathsf{L}}$ reducibility. This is because, assuming $\mathsf{DSPACE}(n) \not\subseteq \mathsf{io\text{-}SIZE}^{\mathsf{MKTP}^A}(2^{\epsilon n})$, any $\leq_{\mathrm{T}}^{\mathsf{L}}$ reduction to MKTP can be simulated by a logspace-uniform $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ reduction to MKTP. (To see this, note that, by Theorem 15, MKTP is hard for DET under this class of reductions, and hence each of the logspace-computable (nonadaptive) queries can be computed using oracle gates for MKTP, and similarly the logspace computation that uses the queries can also be simulated using MKTP. Similar observations arise in [7].)

▶ **Corollary 27.** *If either of* MKTP *or* MCSP *is hard for* NP *under logspace-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions (*P*-uniform* $\leq_{\mathrm{T}}^{\mathsf{AC}^0}$ *reductions), then* $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ *or* NEXP = PSPACE *(*NEXP = EXP*, respectively).*

6 Conclusions and Open Questions

Conclusions. At a high level, we have advanced our understanding about MCSP and MKTP in the following two respects:

1. On one hand, under a very weak cryptographic assumption, the problem of approximating MCSP or MKTP is indeed NP-intermediate under *general* types of reductions when the approximation factor is quite *huge*. This complements the work of [27] for very *restricted* reductions.
2. On the other hand, if the gap is *small*, MKTP is DET-hard under nonuniform NC^0 reductions (contrary to previous expectations). This suggests that nonuniform reductions are crucial to understanding hardness of MCSP. While there are many results showing that NP-hardness of MCSP under *uniform* reductions is as difficult as proving circuit lower bounds, can one show that MCSP is NP-hard under P/poly reductions (without proving circuit lower bounds)?

Open Questions. It should be possible to prove unconditionally that MCSP is not in $\mathsf{AC}^0[2]$; we conjecture that the hardness results that we are able to prove for MKTP hold also for MCSP.

We suspect that it should be possible to prove more general results of the form "If MCSP^A is hard for class $\mathcal{C}$, then so is MKTP^A". We view Theorem 17 to be just a first step in this direction. One way to prove such a result would be to show that MCSP^A reduces to MKTP^A, but (with a few exceptions such as $A = \mathsf{QBF}$) no such reduction is known. Of course, the case $A = \emptyset$ is the most interesting case.

Is MKTP hard for P? Or for some class between DET and P? Is it more than a coincidence that DET arises both in this investigation of MKTP and in the work of Oliveira and Santhanam on MCSP [29]?

Is there evidence that Gap_ϵMCSP has intermediate complexity when ϵ is a fixed constant, similar to the evidence that we present for the case when $\epsilon(n) = o(1)$?

Acknowledgments

We thank Ryan Williams, Rahul Santhanam, Salil Vadhan, and Prashant Nalini Vasudevan for helpful discussions.

References

1 Manindra Agrawal. The isomorphism conjecture for constant depth reductions. *Journal of Computer and System Sciences*, 77(1):3–13, 2011.

2 Manindra Agrawal, Eric Allender, and Steven Rudich. Reductions in circuit complexity: An isomorphism theorem and a gap theorem. *Journal of Computer and System Sciences*, 57(2):127–143, 1998.

3 E. Allender, V. Arvind, R. Santhanam, and F. Wang. Uniform derandomization from pathetic lower bounds. *Philosophical Transactions of the Royal Society, Series A*, 370:3512–3535, 2012. An earlier version appeared in RANDOM 2010. See also the comment number one, posted on ECCC TR10-069.

4 E. Allender, H. Buhrman, M. Koucký, D. van Melkebeek, and D. Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35:1467–1493, 2006.

5 E. Allender, Joshua Grochow, and Cristopher Moore. Graph isomorphism and circuit size. Technical Report TR15-162, Electronic Colloquium on Computational Complexity, 2015.

6 E. Allender, M. Koucký, D. Ronneburger, and S. Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*, 77:14–40, 2010.

7 E. Allender and M. Ogihara. Relationships among PL, #L, and the determinant. *RAIRO - Theoretical Information and Application*, 30:1–21, 1996.

8 Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. In *Mathematical Foundations of Computer Science (MFCS)*, volume 8635 of *Lecture Notes in Computer Science*, pages 25–32. Springer, 2014.

9 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPIcs*, pages 21–33. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.STACS.2015.21.

10 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *Journal of the ACM*, 57:14:1 – 14:36, 2010.

11 Eric Allender, Michal Koucký, Detlef Ronneburger, Sambuddha Roy, and V. Vinay. Time-space tradeoffs in the counting hierarchy. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity*, pages 295–302, 2001. doi:10.1109/CCC.2001.933896.

12 Marco Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Algorithms from natural proofs. In *31st Conference on Computational Complexity, CCC*, volume 50 of *LIPIcs*, pages 10:1–10:24. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.10.

13 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory of Computing*, 9:809–843, 2013. doi:10.4086/toc.2013.v009a026.

14 Oded Goldreich. On promise problems: A survey. In Oded Goldreich, Arnold L. Rosenberg, and Alan L. Selman, editors, *Theoretical Computer Science, Essays in Memory of Shimon Even*, volume 3895 of *Lecture Notes in Computer Science*, pages 254–290. Springer, 2006. doi:10.1007/11685654_12.

15 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

16 J. Håstad. *Computational Limitations for Small Depth Circuits*. MIT Press, Cambridge, MA, 1987.

17 J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182(1):105–142, 1999.

18 J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28:1364–1396, 1999.

19 Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of mcsp and its variants. In *32nd Conference on Computational Complexity, CCC*, LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. to appear.

20 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *31st Conference on Computational Complexity, CCC*, LIPIcs, pages 18:1–10:20. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

21 R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002.

22 R. Impagliazzo and A. Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *ACM Symposium on Theory of Computing (STOC) '97*, pages 220–229, 1997.

23 V. Kabanets and J.-Y. Cai. Circuit minimization problem. In *ACM Symposium on Theory of Computing (STOC)*, pages 73–79, 2000.

24 A. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

25 Richard E. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1):155–171, 1975.

26 Richard E. Ladner and Nancy A. Lynch. Relativization of questions about log space computability. *Mathematical Systems Theory*, 10:19–32, 1976. doi:10.1007/BF01683260.

27 Cody Murray and Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *30th Conference on Computational Complexity, CCC*, volume 33 of *LIPIcs*, pages 365–380. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. URL: http://dx.doi.org/10.4230/LIPIcs.CCC.2015.365, doi:10.4230/LIPIcs.CCC.2015.365.

28 N. Nisan and A. Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

29 I. Oliveira and R. Santhanam. Conspiracies between learning algorithms, circuit lower bounds and pseudorandomness. 2016.

30 Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Structure*, pages 133–138. IEEE Computer Society, 1991. doi:10.1109/SCT.1991.160253.

31 Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *Second Israel Symposium on Theory of Computing Systems (ISTCS)*, pages 3–17. IEEE Computer Society, 1993. doi:10.1109/ISTCS.1993.253489.

32 A. Razborov and S. Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:24–35, 1997.

33 A. A. Razborov. Lower bounds on the size of bounded depth networks over a complete basis with logical addition. *Matematicheskie Zametki*, 41:598–607, 1987. In Russian. English translation in *Mathematical Notes of the Academy of Sciences of the USSR* 41:333–338, 1987.

34 Michael Rudow. Discrete logarithm and minimum circuit size. Technical Report TR16-23, Electronic Colloquium on Computational Complexity, 2016.

35 R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings 19th Symposium on Theory of Computing*, pages 77–82. ACM Press, 1987.

36 Aravind Srinivasan. On the approximability of clique and related maximization problems. *Journal of Computer and System Sciences*, 67(3):633–651, 2003.

37 Jacobo Torán. On the hardness of graph isomorphism. *SIAM Journal on Computing*, 33(5):1093–1108, 2004. doi:10.1137/S009753970241096X.

38 Salil P. Vadhan. An unconditional study of computational zero knowledge. *SIAM Journal on Computing*, 36(4):1160–1214, 2006. doi:10.1137/S0097539705447207.

39 Vinod Vaikuntanathan and Prashant Nalini Vasudevan. Secret sharing and statistical zero knowledge. In *21st International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, volume 9452 of *Lecture Notes in Computer Science*, pages 656–680. Springer, 2015. doi:10.1007/978-3-662-48797-6_27.

40 H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag New York Inc., 1999.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il